TITLE: Application of Micro Genetic Algorithms and Neural Networks for Airfoil Design Optimization

Application of Micro Genetic Algorithms and Neural Networks for Airfoil Design Optimization

Daniel C.M. Tse and **Louis Y.Y. Chan**

Institute for Aerospace Research
Aerodynamics Laboratory
National Research Council Canada
Ottawa, Ontario, Canada K1A 0R6

Abstract

Genetic algorithms are versatile optimization tools suitable for solving multi-disciplinary optimization problems in aerodynamics where the design parameters may exhibit multi-modal or non-smooth variations. However, the fitness evaluation phase of the algorithms casts a large overhead on the computational requirement and is particularly acute in aerodynamic problems where time-consuming CFD methods are needed for evaluating performance. Methods and strategies to improve the performance of basic genetic algorithms are important to enable the method to be useful for complicated three-dimensional or multi-disciplinary problems. Two such methods are studied in the present work: micro genetic algorithms and artificial neural networks. Both methods are applied to inverse and direct airfoil design problems and the resulting improvement in efficiency is noted and discussed.

1 Introduction

Aerodynamic design problems in the transonic regime are characterized by multi-modal topology in the design parameter space with possibilities of non-smooth variations. An example showing the multi-modal variations was constructed by Obayashi and Tsukahara (1997) who also pointed out that the presence of shock waves in transonic flow problems could introduce jump discontinuities in the design parameter space. The ability of genetic algorithms (GAs) to handle non-smooth topology and overcome local critical points in its search for a global optimum makes the method ideal for use in aerospace design optimization.

Although the ease of implementing GAs makes the method attractive, the use of the algorithm is hampered by the large number of fitness evaluations which is particularly costly for aerodynamic design problems. For such problems computational fluid dynamics (CFD) is often needed to provide aerodynamic data for evaluating performance. Thus, it is important to explore methods to improve the efficiency of GAs for on-going extension to three-dimensional applications and multi-disciplinary optimization (MDO).

Two-dimensional airfoil design problems provide a logical and economical means to explore strategies that have the potential to improve the performance of basic GAs. A special GA was designed by Zhu and Chan (1998) to deal specifically with the geometrical aspects of the problem; namely the positions and connectivity of the line segments that make up an airfoil profile. The method directly manipulates the coordinates of the airfoil surface to effect favorable changes, and has been demonstrated to show better efficiency than a standard binary coded GA for a number of two-dimensional examples that deal with geometrical shapes. However, the overall computational requirement is still high and the present work aims to explore methods to further improve the performance.

Two methods to improve the performance of genetic algorithms are studied in this paper, the first is the use of micro genetic algorithms (μGAs) and the second is the application of artificial neural networks (ANNs). The μGA strategy was derived by Goldberg (1989) and Krishnakumar (1989) to explore the use of small population sizes in genetic algorithm applications. Reeves (1993) also showed that for binary coding, a small population size is sufficient to reach the entire search space by cross-over alone. With a small population there will be rapid convergence to a possible sub-optimal solution, and the effective use of μGAs is to repeatedly generate new population members as soon as a measure of convergence has been achieved in a cycle of GA operation.

The implementation of the μGA concept is simple, and the method has been demonstrated to yield marked improvement over conventional large-population GAs. The use of μGAs can be found in Roy et al. (1997), Xiao and Yabe (1998), Abu-Lebdeh and Benekohal (1999), Chang et al. (1999), and the references contained therein. All the studies cited used binary-coded chromosomes, with applications ranging from process engineering to traffic scheduling. Although the range of application of μGAs is becoming extensive, their applicability has yet to be explored fully, and is certainly needed for aerospace design problems where computational requirement is enormous. Also there has not been much demonstration of μGAs for problems that use real number coding and our study provides information in this regard.

Paper presented at the RTO AVT Symposium on "Aerodynamic Design and Optimisation of Flight Vehicles in a Concurrent Multi-Disciplinary Environment", held in Ottawa, Canada, 18-21 October 1999, and published in RTO MP-35.

The second GA improvement strategy studied here is the use of artificial neural networks to supplement detailed computational analysis to provide fitness information for GAs to process. A properly trained ANN can provide fast and inexpensive data to assist engineering designs as demonstrated by Batill and Swift (1993) and Rai and Madavan (1998). The application of ANNs as approximate fitness evaluation tools for GAs, though suggested often, had seldom been put to practice. The combination of ANNs and GAs has been applied mainly for the construction of optimized neural networks through GA-based optimization techniques (Whitley (1995)).

Few applications of ANNs to GA processing have been reported. Podlena and Hendtlass (1998) used an ANN to classify regions in the search space to guide a GA to direct its search to high fitness regions. While details were presented on applications to the standard test cases given by De Jong (1975), only a little information was given for a more complex commercial application on aircraft trajectory planning.

Lingireddy (1998) presents a use of an ANN to provide quantitative data for GA search. In the study for a two-dimensional problem of aquifer parameter identification, GA coupled with finite element analysis (FEA) was found to be effective in estimating hydraulic conductivity but was computationally expensive. A feed-forward neural network was constructed for approximate function evaluation to save on computing cost. The simulation results show it was practical to use ANN and FEA within a genetic algorithm, but the study did not give any quantitative indication of reduction in computational effort.

To fulfill our second objective, an ANN is used as a quantitative alternative to provide aerodynamic coefficients for fitness evaluation so as to reduce the overall computational time of GA operation. The CFD results obtained during part of the GA operation are used to train the ANN in order to construct the response surface representing the design parameter space. With the topology of the design parameter space accurately represented, the fitness information can be obtained readily.

2 Optimization Examples

Two sets of numerical experiments are considered. The first is an inverse design problem where a target pressure distribution around an airfoil is specified and the airfoil shape is obtained by a GA operation to produce a profile that gives the desired pressure distribution. The second is a drag minimization problem with the constraint that the lift is fixed at a specified value.

The airfoil is assumed to operate in the transonic regime at Mach number 0.8 and angle of attack of 1.25°. For the first experiment the target pressure distribution is that of the NACA 0012 airfoil, and the NACA 64A410 is used to derive the starting airfoil samples. For the second experiment, both NACA 64A410 and NACA 0012 are modified to reduce the drag coefficient C_D at fixed values of the lift coefficient C_L which are set to be 0.90 and 0.31 respectively for the two airfoils.

3 Coding Representation of Airfoils

Genetic algorithms work on a coding of design variables representative of the problem; therefore, efficient and effective parametric representation for airfoils is important for the success of the algorithm. For GA applications, airfoils should be represented by as small a parameter set as possible so the search space can be kept small. Also the smoothness of the profile forming an airfoil should be controllable to avoid inadmissible shapes.

Although less amenable to theoretical analysis, real number coding is a preferred choice for airfoil design problems. For such problems, multiple design parameters are involved and these parameters take values over different real number ranges. The use of real number coding eliminates the needs for coding and decoding of design parameters and can provide arbitrarily high resolution to the values of the parameters.

Zhu and Chan (1998) formulated a genetic algorithm for airfoil design problems that used the coordinate points on the airfoil as control variables. Although the underlying geometrical nature of the algorithm allows for the design of efficient GA operators for improved fitness evolution, the large number of design parameters causes difficulties for the mapping of the search space. The coding scheme uses over 100 control variables for a typical airfoil design study, and this number is significantly large to present difficulties in training an accurate ANN as observed by Tse and Chan (1999).

Open B-spline representation of airfoils is used in the present study as only a small number of control points are needed to represent satisfactorily a typical airfoil profile. Other useful advantages of B-spline are the smoothness of the profile can be specified explicitly and the information on concavity is readily available.

In open B-spline representations, the control nodes at the beginning and terminating ends coincide with the end-points of the resulting curve. Because the nose and tail points of an airfoil profile are fixed, the profile is split into a top half and a bottom half and a B-spline curve is used to represent each half.

To fit a B-spline curve to a prescribed profile, the order and number of nodes for the B-spline are specified first and then the resulting coordinates of the nodes are determined by a least square solution method. The B-spline fitting to the NACA 64A410 airfoil is shown in Figure 1. The order of the B-spline curve is six and a total of 10 control nodes are specified to fit each of the top and bottom profiles. A smaller number of B-spline nodes could also yield satisfactory results, but no effort was made to reduce the number of control nodes to an optimum. Details of the theory and numerical algorithms are presented in Rogers and Adams (1990).

4 Genetic Algorithm Operation

Each B-spline control node is considered a gene and the set of ordered control nodes describing a complete airfoil profile is considered a chromosome for GA operation. In subsequent discussion the fitness of a chromosome refers to the aerodynamic performance of an airfoil profile. Manipulation of the locations of the control nodes to obtain an optimal airfoil is the goal of GA operation and the various components of genetic operations are described in the following sections.

4.1 Fitness Evaluation

Fitness evaluation is the basis for the GA search and selection procedure and is usually the most costly part of the algorithm. An Euler flow solver is used as the primary means for obtaining aerodynamic information. This information is made available to train an ANN which then acts as a source for aerodynamic data for evaluating fitness. Detailed discussion of the CFD method and the implementation of ANN is discussed in following sections.

The numerical values of the fitness are also used to differentiate admissible profiles from inadmissible ones. To maintain physical realism, the airfoils should not have too many inflection points where concavity changes sign. If an airfoil exhibits too many inflection points, the fitness for that airfoil is assigned a very low value to ensure it has a low chance of surviving the genetic algorithm.

The fitness values for a generation of chromosomes are ranked and scaled linearly and uniformly. This scaling is necessary to prevent high fitness chromosomes from becoming overly dominant.

4.2 Selection and Population Replacement

Parents are chosen based on the Roulette wheel method where the probability of a chromosome being chosen is proportional to its fractional fitness value. Each pair of parents produces one offspring, and after a new population is produced, their fitness is compared to that of the parent generation and the best members are assigned to be the new generation. This elitist strategy has been observed to yield good performance.

4.3 Cross-over

A simple one-point cross-over scheme is applied. Given two parental chromosomes, an arbitrary cutting point is chosen where the genes positioned after the cutting point are interchanged. The smoothness of the resulting airfoil profiles is guaranteed by the B-spline formulation. The probability of cross-over is set at 80%, and smaller values tend to worsen the performance.

4.4 Mutation

Mutation is carried out by randomly selecting a gene and then changing its value by a random amount chosen within a prescribed range. Let $\{B_1, \ldots, B_{10}\}$ be part of a chromosome that describes one side of an airfoil, with B_1 and B_{10} being the fixed nodes at the leading and trailing edges, respectively, as illustrated in Figure 2. For a gene chosen to undergo mutation (B_5 in Figure 2), its value will be altered by an amount (Δx, Δy) where

$$\Delta x = r_x \cdot \Delta x_{\max}$$

$$\Delta y = r_y \cdot \Delta y_{\max}$$

where r_x and $r_y \in (-0.5, 0.5)$ are random numbers distributed uniformly within the range. In this study $\Delta x_{\max}$ and $\Delta y_{\max}$ are both set to be 1% of chord.

As the above changes are applied to the selected control node, its neighboring control nodes are also adjusted so that the change in slope and curvature of the airfoil profile will not be too abrupt. The changes in the neighboring nodes ($B_2 \ldots B_4$, $B_6 \ldots B_9$) are done in proportion to the relative position from the node where the main change is effected as illustrated in Figure 2. The two end nodes B_1 and B_{10} are left unchanged.

As noted in Obayashi et al. (1998) the mutation probability for a GA using real number coding should be set at a higher value than when binary coding is used. The reason is that in binary coding a change in a single bit can effect significant change in the value of the design variable, but in real number coding a similar change has a lesser effect. Thus a higher mutation probability is justified as a means to enable the algorithm to search the design space thoroughly. The value of mutation probability is set to be 0.8 and lower values give degraded performance.

5 Micro GA Strategy

As discussed earlier, although μGAs have been demonstrated to provide faster convergence than regular GAs when applied to a range of engineering problems, their use has still been quite limited and apparently has not been tried on resource-intensive aerospace optimization problems. An objective of this study is to examine the applicability of μGAs to aerospace design problems where any reduction in computational effort would translate into significant saving in computer resources.

The efficiency of μGAs results from the use of small populations which leads to more rapid convergence and the frequent re-generation of random population members to ensure diversity during the search process. In this study, the μGA is implemented as follows. One cycle of μGA consists of evolving a small population of chromosomes of size *npsize* based on the GA operations described earlier for a fixed number of generations *ngen*. At the end of this cycle, the chromosome with the highest fitness is identified. Then (*npsize* − 1) chromosomes are generated based on mutating the fittest chromosome, and these newly generated chromosomes, together with the fittest individual from the previous μGA cycle, constitute a new population ready for the next stage of μGA operation. This cycling process continues until a desired level of performance criterion is achieved.

6 Methods for Evaluating Fitness

Fitness evaluation through CFD solution is accurate but costly. Hence, supplementary ANN prediction is exploited to provide the required fitness data. Details of the two methods are described below.

6.1 Direct CFD Solution

An efficient CFD solver is needed to yield aerodynamic data quickly for GA processing. A two-dimensional Euler flow solver is used as a compromise between physical realism and computational efficiency.

Central differencing scheme with second and fourth-order artificial dissipation terms is used. A 5-stage Runge-Kutta explicit scheme is used for time marching and the dissipation terms are evaluated at the 1st, 3rd and 5th stages. The optimal values for the coefficients used in the Runge-Kutta scheme are selected according to the suggestions by Jameson et al. (1981). To accelerate convergence a 3-level W-cycle multigrid combined with implicit residual smoothing is used.

For the calculation, an O-type mesh is constructed around the airfoil and far field boundaries are set at 50 chords away to minimize the boundary influence on the solution. There are 160 grid cells distributed around the airfoil and 32 cells along the direction normal to the airfoil surface. The multigrid solver used provides very fast convergence. Details on the solver performance can be found in Zhu and Chan (1998).

6.2 ANN-CFD Coupled Fitness Evaluation

An important consequence of the fact that a GA only needs function values during the optimization search is that the method can tolerate approximate function evaluation more readily than gradient-based methods. Therefore, as stated in Periaux et al. (1995), GAs are robust with respect to noise and accommodate well with less precise solutions. Additional robustness also results from the condition that convergence of the algorithm relies on the entire population and not on a single individual. This robustness quality could be exploited with the help of ANNs, which can construct an accurate response surface for the search space.

Applications of ANNs as response surfaces for aerodynamic data estimation and extraction have been reported by Chan and Zhu (1999) and Greenman and Roth (1999). The results indicate that a suitably trained ANN can provide accurate nonlinear interpolation for aerodynamic data.

The coupling between ANN and CFD for fitness evaluation proceeds as follows. First, specify those generations as *igen-cfd* where fitness information is obtained by the direct CFD method, and those other generations as *igen-ann* where approximate ANN evaluation is used. For each of the generations labeled *igen-cfd*, CFD is used for fitness evaluation and the results are stored for training the ANN. ANN training is commenced as soon as the CFD evaluation phase is over. Data for the whole generation of chromosomes become training cases for the ANN, and the number of training cases is the number of chromosomes in a population.

Next, for each of the generations labeled *igen-ann*, the aerodynamic data for the chromosomes are obtained by the trained ANN. Because ANN results have lower fidelity compared to CFD predictions, the ANN results need scrutiny. Let *fmax* be the maximum fitness predicted by the CFD method; then if the ANN prediction for a chromosome has fitness greater than 90% of *fmax*, the fitness is re-calculated using CFD.

7 Neural Network Construction and Performance

The ANN used in this study is a single layer version of the Multilayer Perceptron network as shown in Figure 3. The neural network is designed to take the *(x,y)* coordinates of the B-spline control nodes as input and the pressure values around the airfoil surface are the output. Since there are 20 B-spline control nodes for an airfoil profile, the number of input nodes is 40 to account for the x and y values. Because 160 grid locations are defined in the CFD calculation for the surface pressure, the number of output nodes is the same at 160. Ten neurons are used in the hidden layer and numerical experimentation shows that the ANN results are not too sensitive to the number of hidden layer neurons used.

The training of the ANN is done by the well-known backpropagation algorithm. Techniques such as appropriate scaling of the input and output signals as suggested in Haykin (1994) are applied to accelerate the convergence.

When an ANN is used within a genetic algorithm, the training of the ANN is continued for the entire duration of the GA operation. The weights obtained from one set of training data are kept and are used as initial values to train the next set of data. This method reduces the training errors for a fixed amount of training epochs and enables a more comprehensive construction of the response surface for the search space.

Because the CFD flow solver is very efficient for the current applications, the training time required for the ANN becomes the bottleneck of the computational resource. Upon some experimentation, it was decided that 1000 epochs will be implemented for the neural network training. The choice represents a compromise between accuracy and computing time requirement. Typical prediction results are displayed in Figures 4 and 5, which show the surface pressure distribution for two airfoils obtained during the course of μGA operation.

In general, the ANN predictions are quantitatively reliable, with the shock jump magnitude and position predicted quite accurately for the case shown in Figure 4, but the position was a bit off for the case represented in Figure 5. For the results in Figure 4, the CFD method gives C_D to be 0.082 while the ANN method gives 0.077, an error of 6.1%. For C_L, CFD gives 0.93 while ANN gives 0.89, an error of 4.3%. For the results in Figure 5, the CFD method gives C_D to be 0.078 while the ANN method gives 0.069, an error of 12%. For C_L, CFD gives 0.92 while ANN gives 0.84, an error of 8.7%. This range of accuracy represents what typically can be expected from the ANN predictions. Compared to the results presented in Tse and Chan (1999) where airfoil coordinates were used as design variables, the training time for the ANN is now shorter and the accuracy is higher.

8 Results of Numerical Experiments

For the numerical experiments, all the cases are executed for approximately 10000 fitness evaluations and the resulting performance of different strategies are compared based on the amount of CFD solver calls. For the cases where regular GA is used, a population size of 40 is chosen and 250 generations are computed to attain the desired number of fitness evaluations. The GA parameters are kept fixed in all cases.

8.1 Pressure Distribution Matching

A target pressure distribution is given as $\hat{p}_i$, $i = 1, 2, \ldots, N_g$; where $N_g = 160$ is the number of surface grid cells around the airfoil. Let p_i be the pressure distribution around an airfoil design candidate, the goal is to minimize the following objective function *obj*:

$$obj = \frac{1}{N_g}\sum_{i=1}^{N_g}(p_i - \hat{p}_i)^2$$

The fitness function used in GA processing is defined as the reciprocal of *obj*.

8.1.1 Performance Comparison

Different Parameters for μGA

A parametric study is made to vary the values of *npsize* and *ngen* for μGA operation. The case *npsize* = 10 and *ngen* = 10 is chosen as the base case. The case *npsize* = 5 and *ngen* = 10 is done to examine the effect of reducing the population size. Following the work by Krishnakumar (1989), most applications with μGA use a population size of 5, usually without much investigation.

The work by Abu-Lebdeh and Benekohal (1999) shows that mid-sized populations (*npsize* between 9 and 15) consistently exhibited lower internal variability, defined as the variation in fitness value caused by changes in the initial population. Hence, a comparison of the effect of different *npsize* is desirable. In addition, the case *npsize* = 10 and *ngen* = 20 is tried to observe any changes in the fitness performance when a better convergence is established within each μGA cycle.

Results of the parametric study are shown in Figure 6. It is of interest to observe that for the case *npsize* = 5, the initial performance is far superior to the other two cases, but the performance levels off after the initial stage. This shows it is indeed favorable to use *npsize* = 5 in μGA operation, especially for dynamical modeling problems where real-time solutions are needed. The case with *ngen* = 20 does not provide any performance improvement. For the rest of this study, the base case parameters are used because of the better performance when sufficient GA processing is allowed.

Comparison between μGA and GA

The significant improvement in fitness evaluation of μGA over GA is demonstrated in Figure 7. In the example, μGA is operated for 100 cycles and the GA for 250 generations, making the number of fitness evaluation or CFD solver calls to be about 10000 for both cases. Note that the actual CFD solver calls are slightly less than 10000 because some chromosomes generated do not satisfy the concavity constraints as stated earlier and are rejected. The superiority of μGA is evident: the μGA gives higher fitness prediction during the entire course of the optimization process.

Results with ANN Coupling

Figure 8 shows the fitness performance when ANN is used for part of the fitness evaluation. The implementation of ANN is as follow: during a cycle of μGA with *ngen* = 10, the aerodynamic data for the first five generations are calculated with CFD and the results are used to train the ANN. For the sixth generation CFD is also used but the data are for validating the ANN and not for training. For the remaining four generations the ANN is solely responsible for providing aerodynamic data.

The performance result is plotted with CFD solver calls alone, with the computational effort of ANN training not represented. The purpose is to highlight the computational saving achieved by reducing the overhead on the CFD solver. For the particular type of problems under study and the particular flow solver used, the CPU time for calculating one airfoil is about 3.9 seconds on an SGI Octane computer. In contrast, the CPU time for 1000 epochs of ANN training is about 7.4 seconds, almost twice that of the CFD calculation. However, for more complicated three-dimensional applications, the amount of CFD solver usage would dominate the CPU time consumption.

Figure 9 illustrates the benefit of using a larger training set by comparing the case with 4 generations of data available for training to that with 5 generations. Although the computational requirement is slightly higher for the latter case, a larger training set could give better definition for the underlying response surface and the advantage is reflected in the resulting improvement in the fitness evaluation.

8.1.2 Inverse Design Results

Figure 10 shows the pressure distribution for the target design (NACA 0012) and the initial design (NACA 64A410). The transformation of one profile to the other effects readily as indicated by the fitness results. The final profile obtained by the μGA is very close to the desired profile, as shown in Figure 11.

8.2 Drag Minimization Studies

For drag minimization at fixed lift, the objective function is defined as

$$obj = C_D + \beta \cdot (C_L - C_L^*)^2$$

where the constraint $C_L = C_L^*$ has been incorporated by a penalty function formulation with β being the penalty coefficient, chosen to be 10. The aim is to minimize *obj* and the GA fitness function is defined as obj^{-1}. The value for C_L^* is 0.90 for the NACA 64A410 case and 0.31 for the NACA 0012 case.

8.2.1 NACA 64A410

Figure 12 shows the performance of various strategies for the calculation starting with the NACA 64A410. Again µGA performs better than the regular large population GA, but the improvement is not as marked as in the previous example. When ANN is used, the performance is slightly better. Figure 13 illustrates again that it is beneficial to use a larger training set despite the slightly higher ANN training overhead.

The Mach contours of the original NACA 64A410 and the µGA optimized airfoil are shown in Figures 14 and 15 respectively. The airfoil designed by the µGA has C_L = 0.91 and C_D = 0.0056, a mere 7.4% of 0.076, the C_D value for NACA 64A410.

It is striking to see that without any thickness constraint, the airfoil would evolve into a very thin structure with a large leading edge curvature as shown in Figure 16. The large curvature encourages a rapid flow expansion, and the flattened upper surface then reduces the flow expansion to a near roof-top type distribution with a lower pressure level as shown in Figure 17. Consequently the shock strength is weakened as shown in both Figure 15 and Figure 17.

8.2.2 NACA 0012

Figure 18 shows the performance of µGA for this problem. Again, using ANN for part of the fitness prediction enhances the performance: for the same amount of CFD solver calls, calculation with ANN tends to give similar or better fitness results.

The Mach number distributions for the original NACA 0012 and the µGA optimized airfoil are shown in Figures 19 and 20 respectively. The µGA optimization process essentially eliminates the shock near the aft part of the airfoil. Figure 21 shows a detailed comparison of the profile shapes between the initial and optimized airfoils. Again, the optimization aims to increase the leading edge curvature and flatten both the upper and lower surfaces. The elimination of the shock is also seen in Figure 22, which shows the surface pressure distribution. For the µGA optimized airfoil, C_L is 0.31 while C_D is 0.0012, a mere 5.2% of 0.023, the C_D value for NACA 0012.

9 Conclusion

The examples shown suggest micro genetic algorithms can work well for problems that use real number coding. The performance of micro genetic algorithms is consistently better than regular genetic algorithms, although the improvement is more striking for some cases than others. With plenty of GA processing, the use of mid-sized populations yields higher fitness performance, but for solving real-time dynamical problems, a small population size appears to be more beneficial.

The artificial neural network constructed in this study performs well for the prediction of the surface pressure distribution. The use of B-spline control nodes as design variables instead of airfoil coordinates leads to a smaller network which enhances the performance, both for reducing the time required to train the network and for reducing the errors associated with the prediction. The good quantitative comparison indicates that ANN can realistically replace some of the solver calls in aerodynamic design applications.

Within a genetic algorithm, the use of ANNs can reduce the amount of CFD solver calls needed to achieve the same level of fitness performance. This saving of solver calls is essential for further applications of genetic algorithms to design problems under more complex conditions such as three-dimensionality, more complicated flow physics, and integration with other disciplines.

References

Abu-Lebdeh, G. and Benekohal, R.F. (1999), "Convergence Variability and Population Sizing in Micro-Genetic Algorithms", *Computer-Aided Civil and Infrastructure Engineering*, Vol.14, No.5, pp.321-334.

Batill, S.M. and Swift, R.A. (1993), "Structural Design Space Definition using Neural Networks and a Reduced Knowledge Base", AIAA-93-1034, AIAA/AHS/ASEE Aerospace Design Conference, Feb.16-19, Irvine, CA.

Chan, L. and Zhu, Z. (1999), "Neural Networks Modeling of Aerodynamics and Applications to Store Separation", AIAA-99-0685, 37th AIAA Aerospace Sciences Meeting and Exhibit, Jan. 11-14, Reno, NV.

Chang, C.S., Jiang, W.Q. and Elangovan, S. (1999), "Application of genetic algorithms to determine worst-case switching overvoltage of MRT systems", *IEE Proc.-Electr. Power Appl.*, Vol.146, No.1, pp.81-87.

De Jong, K.A. (1975), "Analysis of the Behavior of a Class of Genetic Adaptive Systems", Ph.D. Dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI.

Goldberg, D.E. (1989), "Sizing Populations for Serial and Parallel Genetic Algorithms", Proceedings of the Third International Conference on Genetic Algorithms, pp.70-79.

Greenman, R.M. and Roth, K.R. (1994), "Minimizing Computational Data Requirements for Multi-Element Airfoils using Neural Networks", AIAA-99-0258, 37th AIAA Aerospace Sciences Meeting and Exhibit, Jan. 11-14, Reno, NV.

Haykin, S. (1994), Neural Networks: A Comprehensive Foundation, Prentice Hall.

Jameson, A, Schmidt, W. and Turkel, E. (1981), "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes", AIAA-81-1259, AIAA 14th Fluid and Plasma Dynamics Conference, June23-25, Palo Alto, CA.

Krishnakumar, K. (1989), "Micro-Genetic Algorithms for Stationary and Non-Stationary Function Optimization", *SPIE Vol.1196, Intelligent Control and Adaptive Systems*, pp.289-296.

Lingireddy, S. (1998), "Aquifer Parameter Estimation using Genetic Algorithms and Neural Networks", *Civil Eng. and Env. Syst.*, Vol.15, pp.125-144.

Obayashi, S. and Tsukahara, T. (1997), "Comparison of Optimization Algorithms for Aerodynamic Shape Design", *AIAA Journal*, Vol.35, No.8, pp.1413-1415.

Obayashi, S., Takahashi, S. and Takeguchi, Y. (1998), "Niching and Elitist Models for MOGAs", Parallel Problem Solving from Nature - PPSN V, Lecture Notes in Computer Science, Springer, Berlin, Germany, pp.260-269.

Periaux, J., Sefrioui, M., Stoufflet, B., Mantel, B. and Laporte, E. (1995), "Robust Genetic Algorithms for Optimization Problems in Aerodynamic Design", Genetic Algorithms in Engineering and Computer Science, edited by G. Winter et al., Wiley Publishers, pp.371-396.

Podlena, J.R. and Hendtlass, T. (1998), "An Accelerated Genetic Algorithm", *Applied Intelligence*, Vol.8, pp.103-111.

Rai, M.M. and Madavan, N.K. (1998), "Aerodynamic Design Using Neural Networks", AIAA-98-4928, 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Sept.2-4, St. Louis, MO.

Reeves, C.R. (1993), "Using Genetic Algorithms with Small Populations", Proceedings of the Fifth International Conference on Genetic Algorithms, pp.92-99.

Rogers, D.F. and Adams, J.A. (1990), Mathematical Elements for Computer Graphics, second edition, McGraw-Hill Publishing Company.

Roy, S., Ghosh, S. and Shivpuri, R. (1997), "A New Approach to Optimal Design of Multi-Stage Metal Forming Processes with Micro Genetic Algorithms", *Int. J. Mach. Tools Manufact.*, Vol.37, No.1, pp.29-44.

Tse, D. and Chan, L. (1999), "Optimization of Airfoil Designs using a Geometric Genetic Algorithm and a Neural Network", Proceedings of the 46th Annual Conference, Canadian Aeronautics and Space Institute, pp.1-8.

Whitley, D. (1995), "Genetic Algorithms and Neural Networks", Genetic Algorithms in Engineering and Computer Science, edited by G. Winter et al., Wiley Publishers, pp.203-216.

Xiao, F. and Yabe, H. (1998), "Microwave Imaging of Perfectly Conducting Cylinders from Real Data by Micro Genetic Algorithm Coupled with Deterministic Method", *IEICE Trans. Electron.*, Vol.E81-C, No.12, pp.1784-1792.

Zhu, Z.W. and Chan, Y.Y. (1998), "A New Genetic Algorithm for Aerodynamic Design Based on Geometric Concept", AIAA-98-2900, 29th AIAA Fluid Dynamics Conference, June 15-18, Albuquerque, NM.

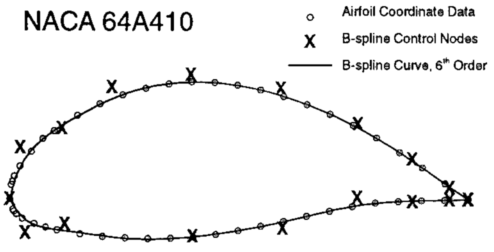

Figure 1: B-spline representation of the NACA 64A410 airfoil; 20 B-spline nodes, 6th order B-spline curve, vertical scale expanded.

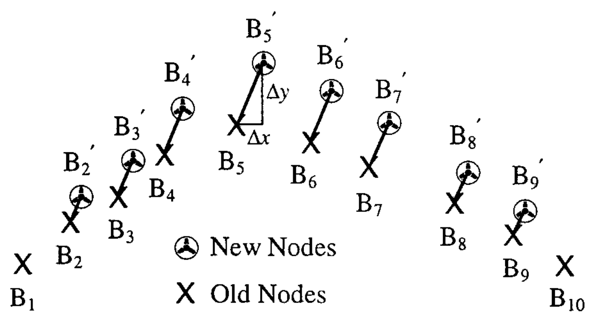

Figure 2: Schematic drawing showing the effect of mutation at the node B_5 and the subsequent smoothing.

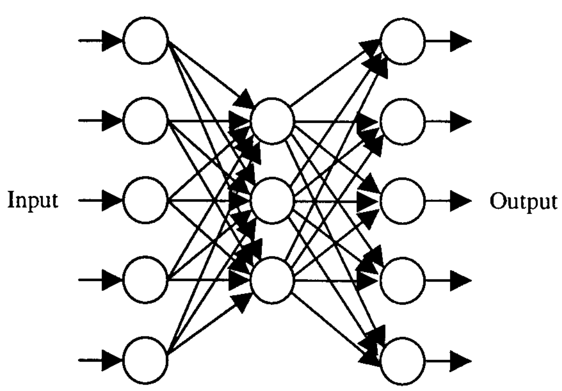

Figure 3: Schematic layout of a Multilayer Perceptron ANN.

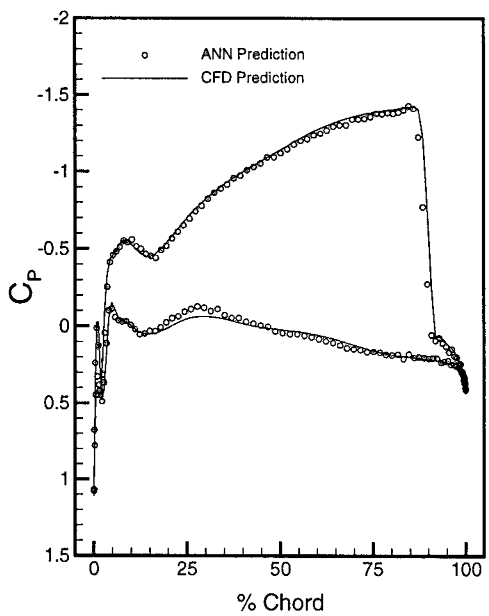

Figure 4: Pressure distribution prediction, CFD vs. ANN results; C_D error = 6.1%, C_L error = 4.3%.

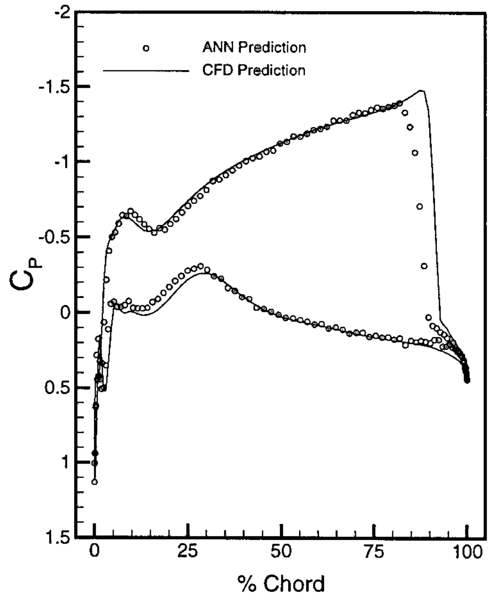

Figure 5: Pressure distribution prediction, CFD vs. ANN results; C_D error = 12%, C_L error = 8.7%.

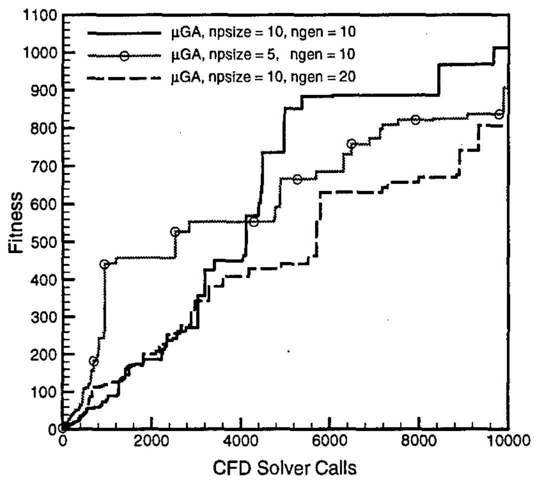

Figure 6: Performance comparison for μGA using various parameters.

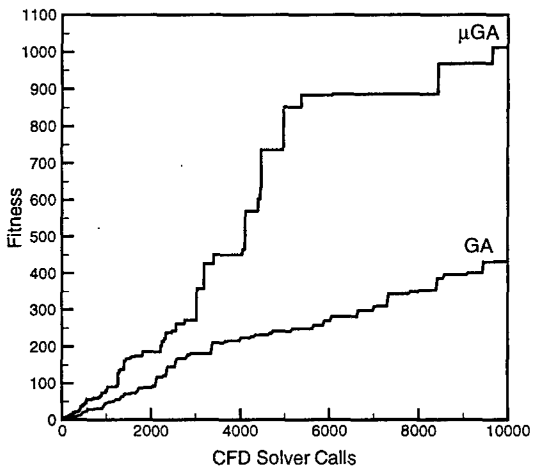

Figure 7: Performance comparison between μGA and GA.

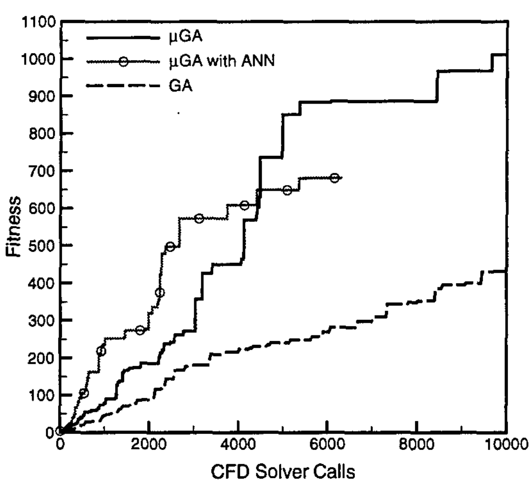

Figure 8: Performance comparison when ANN is used in fitness evaluation.

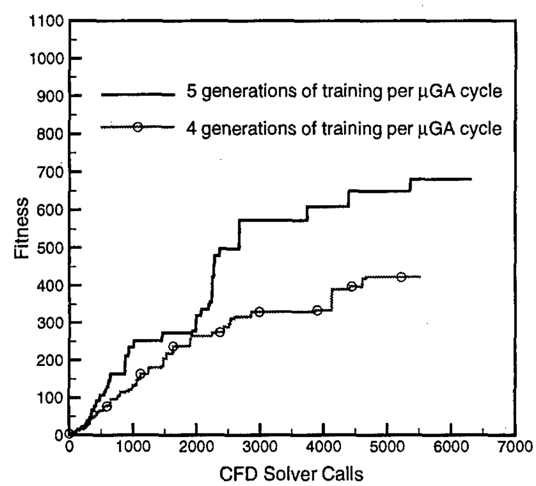

Figure 9: Comparison showing more ANN training sets can be beneficial.

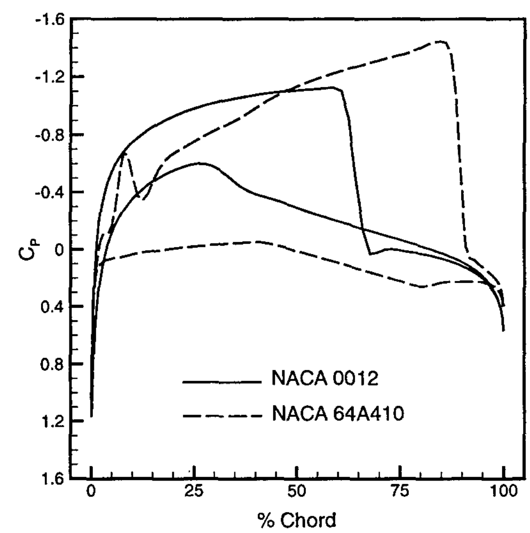

Figure 10: Surface pressure distribution; NACA 64A410 is used as the initial distribution, NACA 0012 is the target. Airfoils operate at Mach 0.8, angle of attack 1.25°.

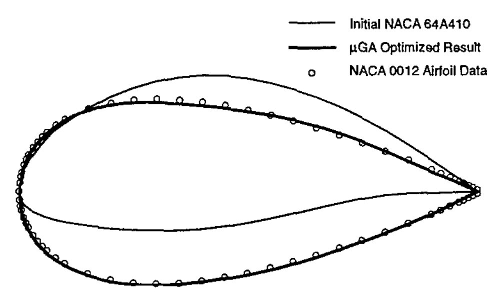

Figure 11: Results of the inverse design optimization showing the initial profile, the target profile and the μGA result, vertical scale expanded.

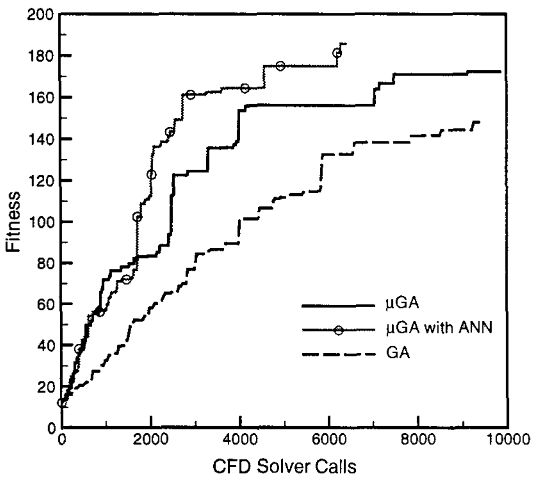

Figure 12: Performance comparison for the NACA 64A410 airfoil drag minimization study.

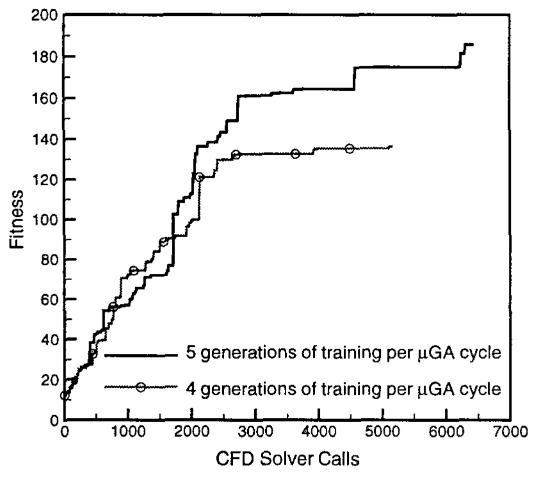

Figure 13: Performance comparison showing the benefit of more training per μGA cycle.

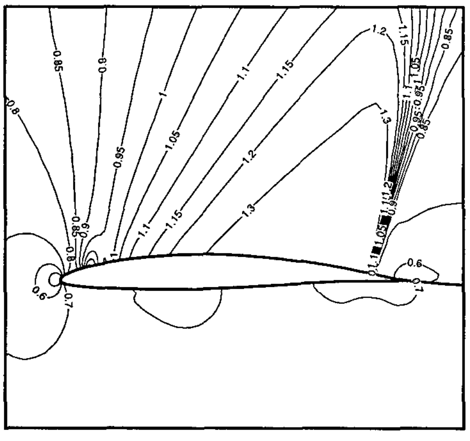

Figure 14: Mach number distribution for the NACA 64A410.

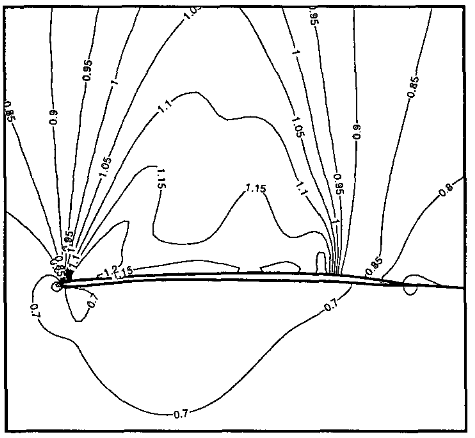

Figure 15: Mach number distribution for the μGA optimized profile.

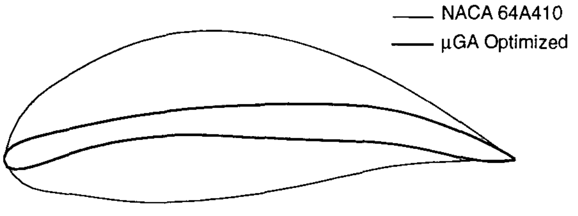

Figure 16: Results of drag minimization showing the initial NACA 64A410 and the optimized profile, vertical scale expanded.

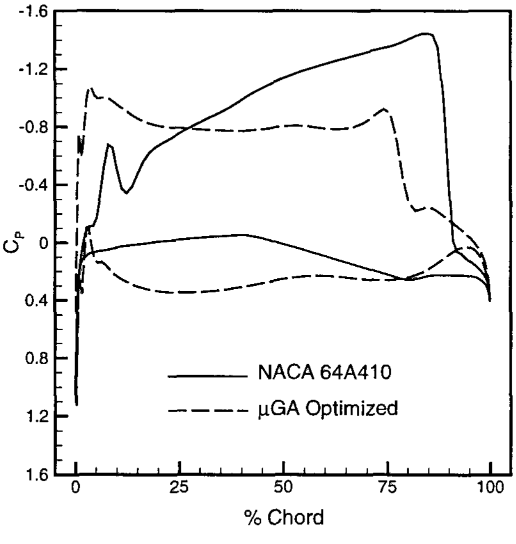

Figure 17: Surface pressure distribution for the initial NACA 64A410 and the μGA optimized profile.

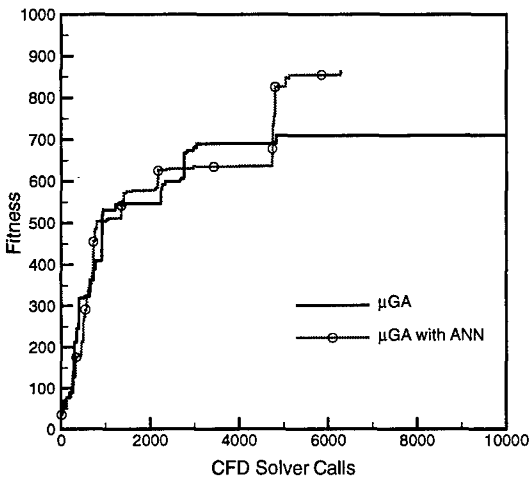

Figure 18: Performance comparison for the NACA 0012 airfoil drag minimization study.

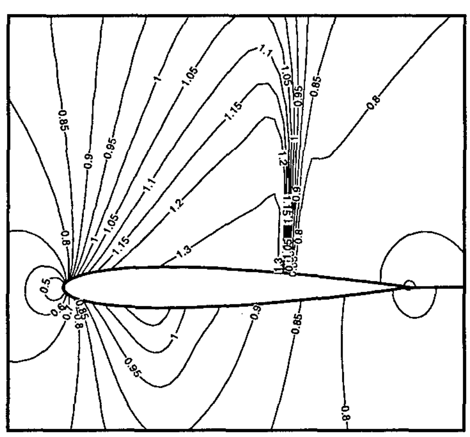

Figure 19: Mach number distribution for the NACA 0012.

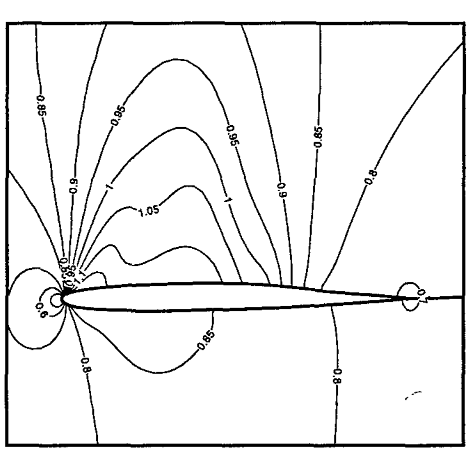

Figure 20: Mach number distribution for the μGA optimized profile.

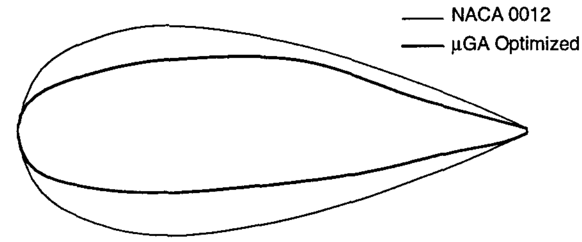

Figure 21: Results of drag minimization showing the initial NACA 0012 and the optimized profile, vertical scale expanded.

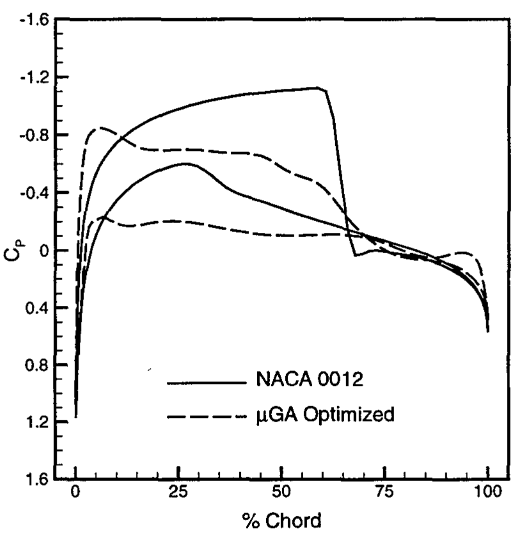

Figure 22: Surface pressure distribution for the initial NACA 0012 and the μGA optimized profile.